1 Strings are Null-terminated Arrays

We have used strings with double quotes—these are constant strings. But what about testing and changing strings? In C, a string is a null-terminated sequence of characters; that is, there is a special character—written '\0'—after the last normal character.

The standard approach for a user-created string is to store it inside a character array. In particular, this means that the array must have size at least 1 more than the length of the string. For example, a string like "happy", which has length 5, is stored in a char array of size at least 6 as:

```
0 1 2 3 4 5 6 onwards
h a p p y \0 irrelevant
```

Constant strings, like the ones we provide to printf, automatically have the null character added to them. But any user-created string must have '\0' explicitly added.

To print a string with printf or read a string with scanf, use %s. Note that scanf is passed just the name of the char array (we’ll see why later)—no ampersand. Also, scanf ignores whitespace before the string and treats a whitespace as the end of string.

You can create your functions to do many things with strings. For standard tasks, there are function in the string library.

2 Creating Your Own String Function

Almost all string functions have a main loop that iterates until the end of the string is detected. For example, here is code to compute the length of a string:

```c
int strlen(char s[]) {
    int x = 0;
    while (s[x] != '\0')
        x=x+1;
    return x;
}
```

It returns 5 when called by

```c
char test[] = "happy";
strlen(test);
```
Or suppose you wanted to convert a string to all capitals:

```c
void toUpperCase(char s[]) {
    int x;
    for( x=0; s[x] != '\0'; x++ ) {
        if( s[x]>='a' && s[x]<='z')
            s[x] += 'A' - 'a';
    }
}
```